UML Diagrams by type

The Unified Modeling Language

The Unified Modeling Language (UML) is a set of 13 different diagram types that may be used to model software systems. It emerged from work in the 1990s on object-oriented modeling, where similar object-oriented notations were integrated to create the UML. A major revision (UML 2) was finalized in 2004. The UML is universally accepted as the standard approach for developing models of software systems. Variants, such as SysML, have been proposed for more general system modeling.

http://software-engineering-book.com/web/uml/

5.1 Context Models

Context models are often used to show the external factors and systems that interact with the system you are modeling.

- **Context Diagram** - not sure of its actual UML name
-

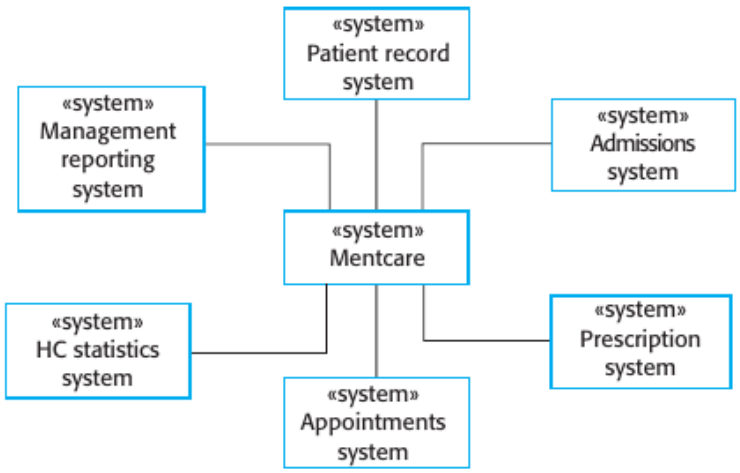

Figure 5.1 The context of the Mentcare system

- **Use Case Diagram**: Used to represent the functionality of a system from an end-user's point of view. It defines the system's boundary and scope and identifies the primary actors and their interactions with the system.

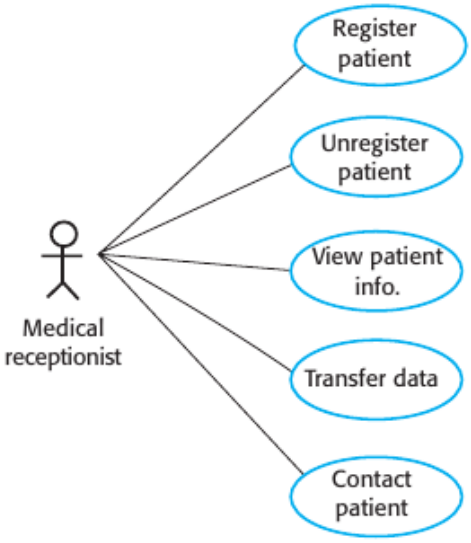

Figure 5.5 Use cases involving the role "Medical receptionist"

Figure 5.3 Transfer-data use case

Mentcare system: Transfer data	
Actors	Medical receptionist, Patient records system (PRS)
Description	A receptionist may transfer data from the Mentcare system to a general patient record database that is maintained by a health authority. The information transferred may either be updated personal information (address, phone number, etc.) or a summary of the patient's diagnosis and treatment.
Data	Patient's personal information, treatment summary
Stimulus	User command issued by medical receptionist
Response	Confirmation that PRS has been updated
Comments	The receptionist must have appropriate security permissions to access the patient information and the PRS.

Figure 5.4 Tabular description of the Transfer-data use case

5.2 Interaction Models

These models capture the dynamic interactions between objects and components.

- **Sequence Diagram**: Used to display the interaction between objects in the chronological order that those interactions occur.

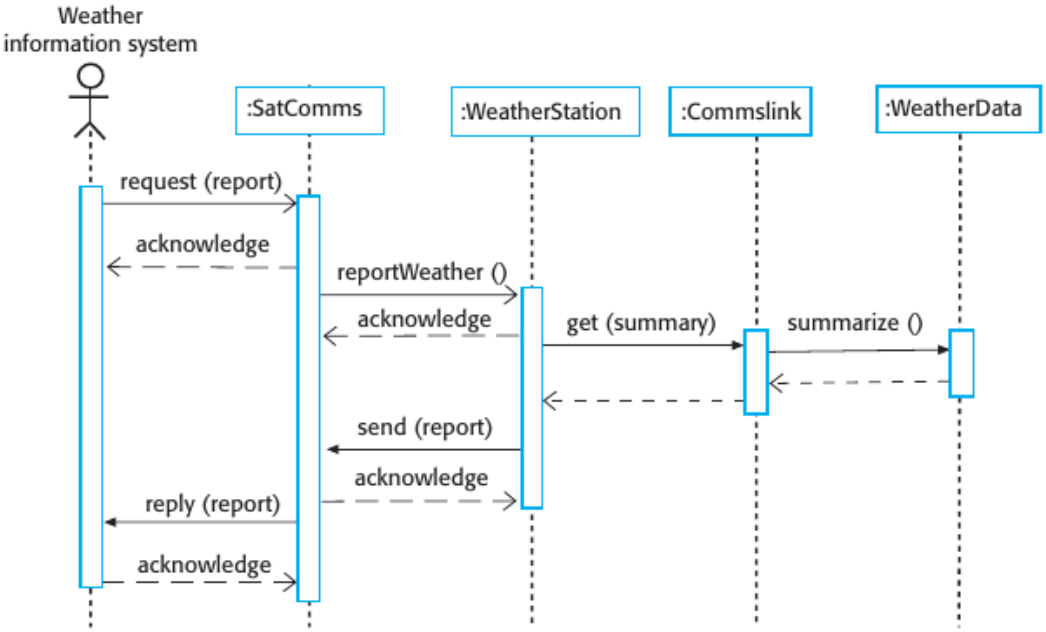

Figure 7.7 Sequence diagram describing data collection

in doing too much modeling. You should not make detailed decisions about the implementation that are really best left until the system is implemented.

Sequence models are dynamic models that describe, for each mode of interaction

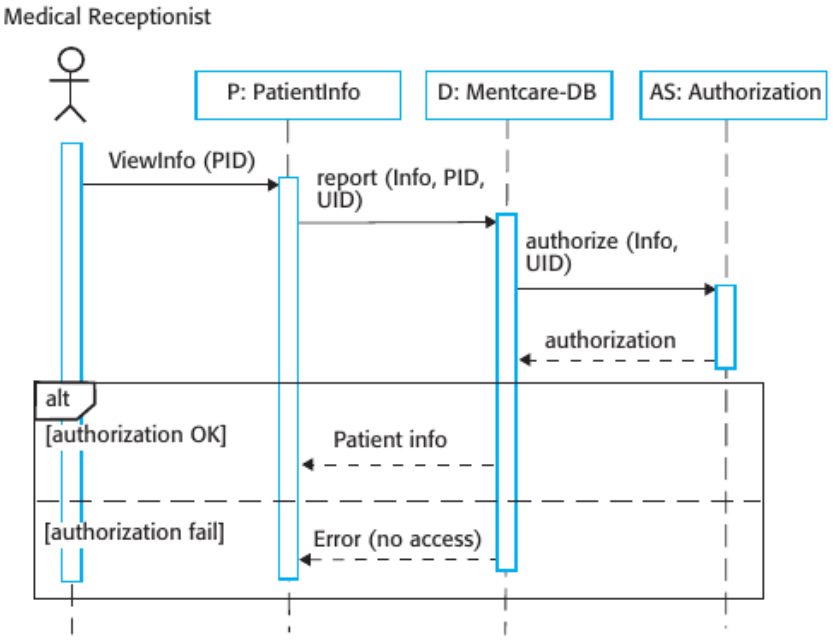

Figure 5.6 Sequence diagram for View patient information

- **Communication Diagram**: Illustrates the interactions between objects or parts in terms of sequenced messages. It provides an overview of the architecture of the system.
- communication diagrams are simply an alternative representation of sequence diagrams.

- **Activity Diagram**: Represents workflows between various system components and activities. It is essentially a flowchart to represent the flow from one activity to another activity.

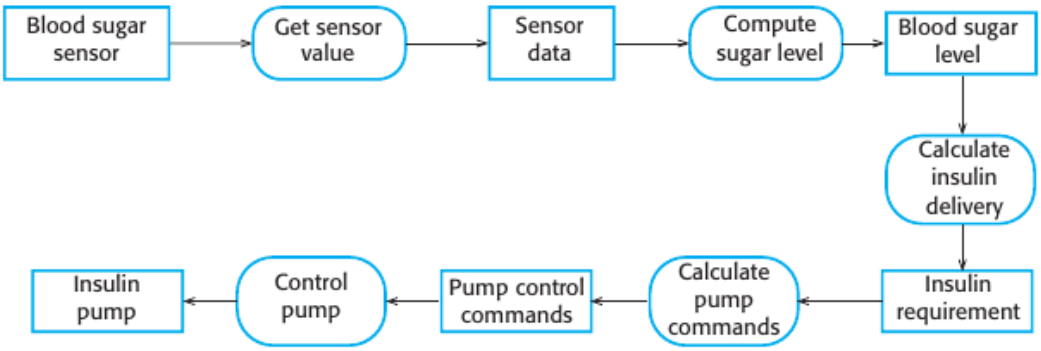

Figure 5.14 An activity model of the insulin pump's operation

processing steps in a system. Data-flow models are useful because tracking and documenting how data associated with a particular process moves through the system help analysts and designers understand what is going on in the process. DFDs are

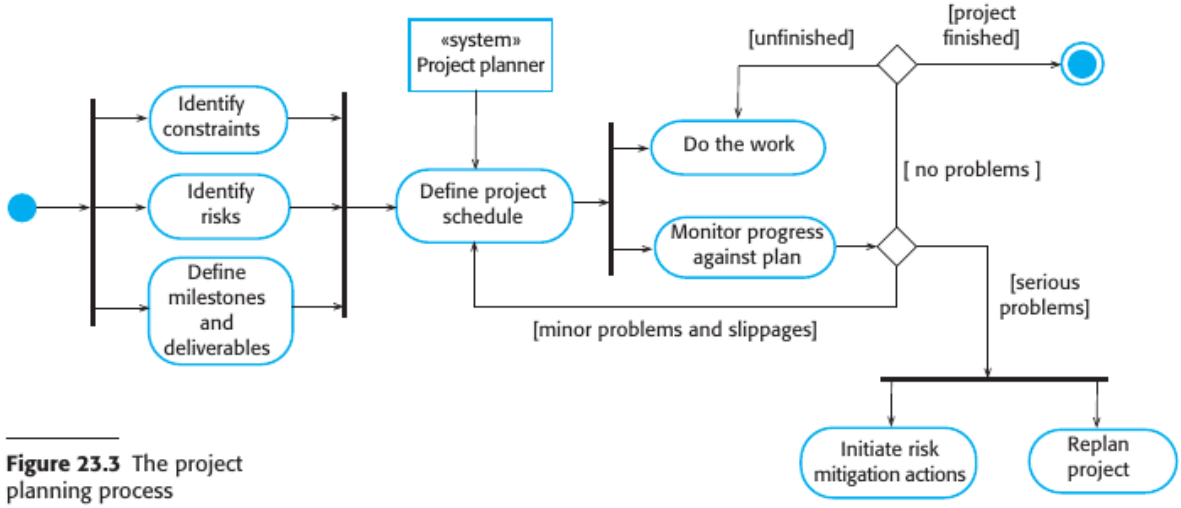

Figure 23.3 The project planning process

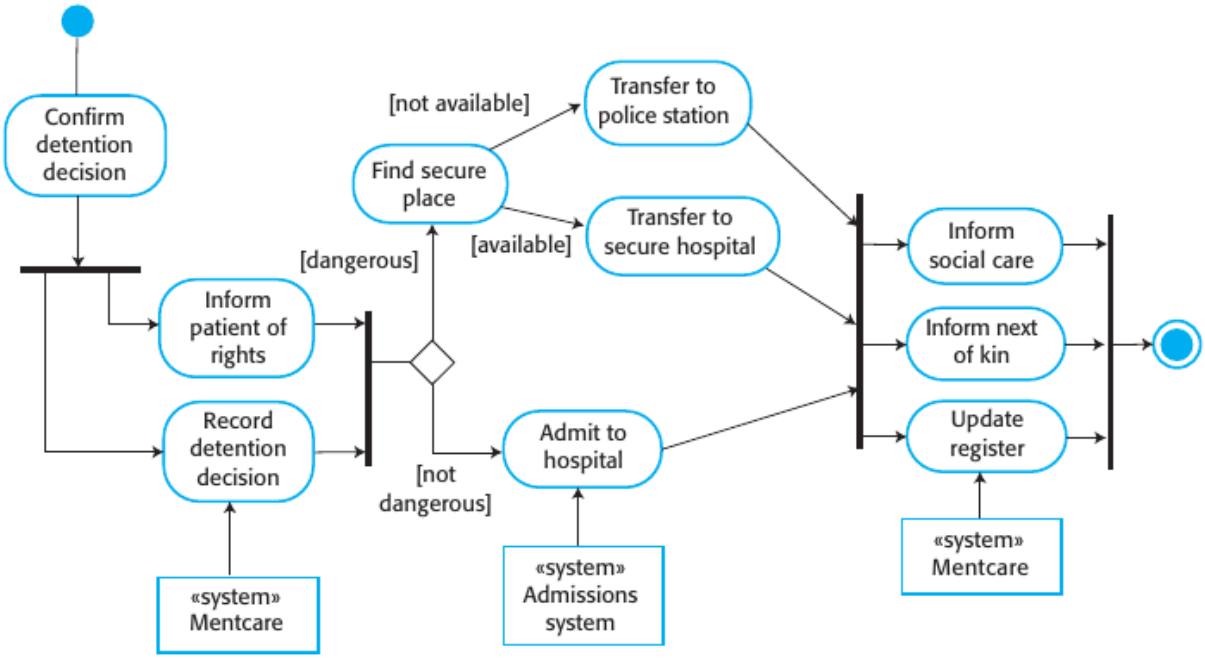

Figure 5.2 A process model of involuntary detention

these relations may affect the requirements and design of the system being defined and so must be taken into account. Therefore, simple context models are used along

5.3 Structural Models

Structural models define the static architecture of a system.

- **Class Diagram**: Describes the structure of a system by showing its classes, their attributes, and the relationships among them.

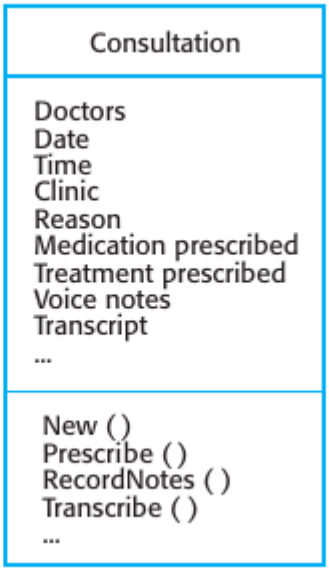

Figure 5.10 A Consultation class

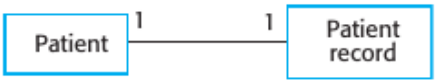

Figure 5.8 UML Classes and association

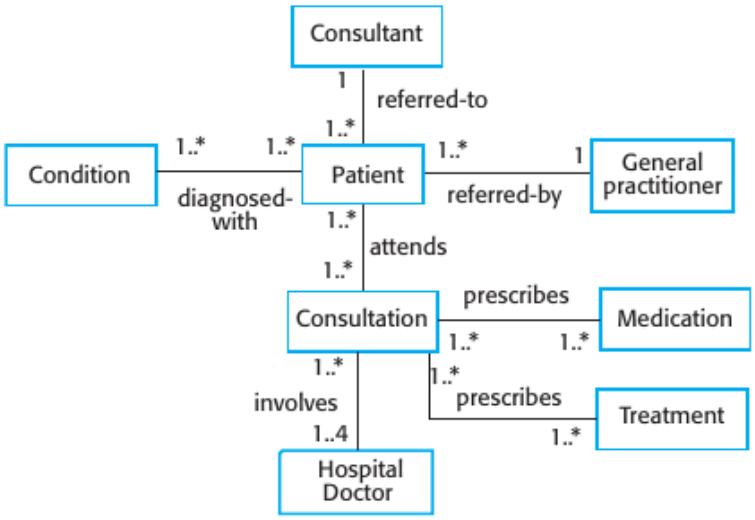

Figure 5.9 Classes and associations in the Mentcare system

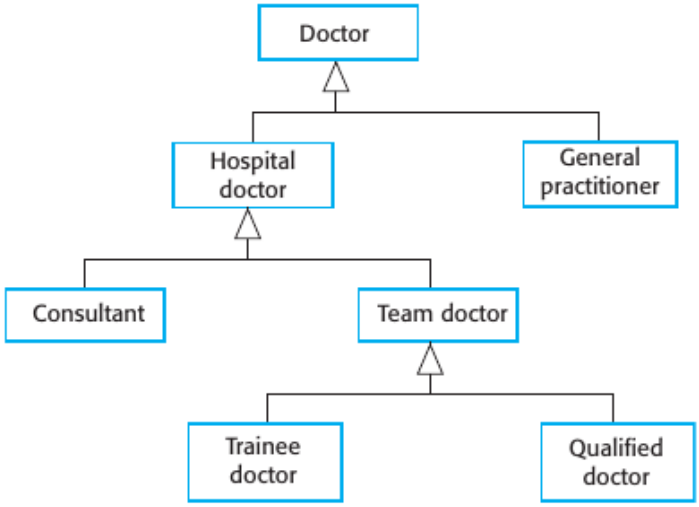

Figure 5.11 A generalization hierarchy

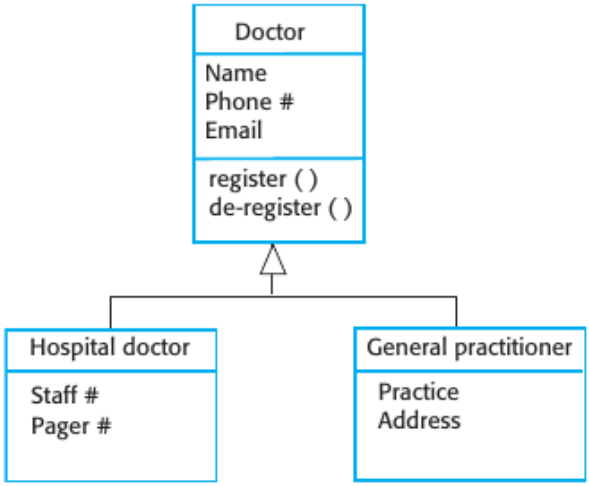

Figure 5.12 A generalization hierarchy with added detail

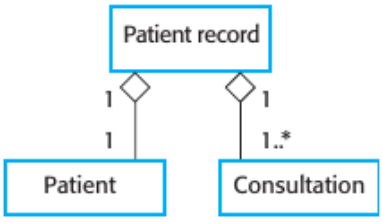

Figure 5.13 The aggregation association

5.4 Behavioral Models

Behavioral models show the dynamic aspects and the behavior of the system components.

- **State Diagram (State Machine Diagram)**: Describes the states an object or interaction may be in, as well as the transitions between states.

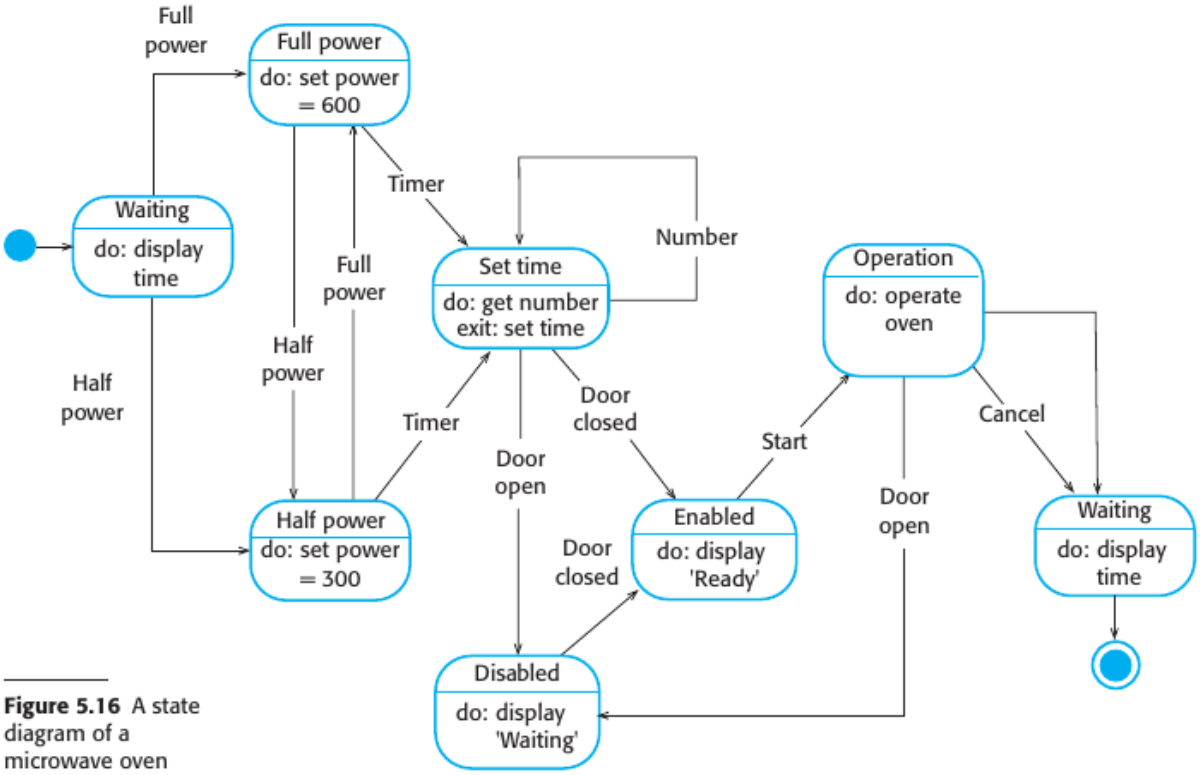

Figure 5.16 A state diagram of a microwave oven

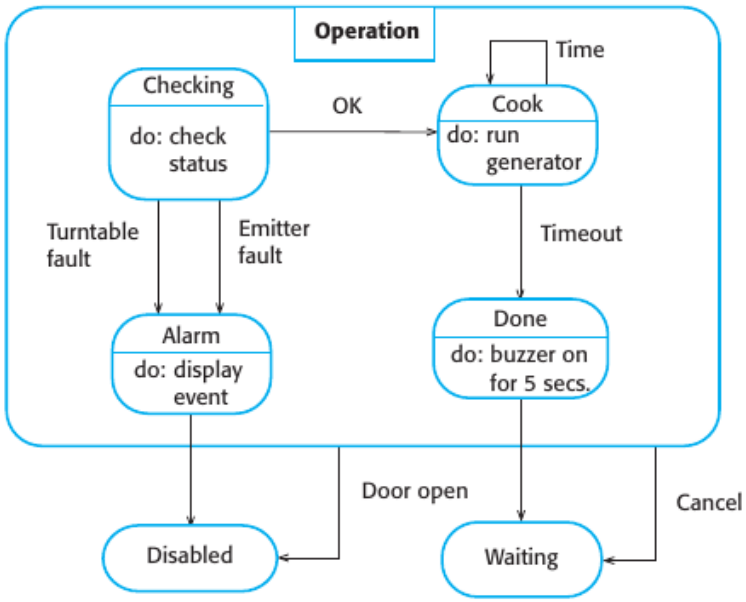

Figure 5.17 A state model of the Operation state

State	Description
Waiting	The oven is waiting for input. The display shows the current time.
Half power	The oven power is set to 300 watts. The display shows "Half power."
Full power	The oven power is set to 600 watts. The display shows "Full power."
Set time	The cooking time is set to the user's input value. The display shows the cooking time selected and is updated as the time is set.
Disabled	Oven operation is disabled for safety. Interior oven light is on. Display shows "Not ready."
Enabled	Oven operation is enabled. Interior oven light is off. Display shows "Ready to cook."
Operation	Oven in operation. Interior oven light is on. Display shows the timer countdown. On completion of cooking, the buzzer is sounded for 5 seconds. Oven light is on. Display shows "Cooking complete" while buzzer is sounding.
Stimulus	**Description**
Half power	The user has pressed the half-power button.
Full power	The user has pressed the full-power button.
Timer	The user has pressed one of the timer buttons.
Number	The user has pressed a numeric key.
Door open	The oven door switch is not closed.
Door closed	The oven door switch is closed.
Start	The user has pressed the Start button.
Cancel	The user has pressed the Cancel button.

Figure 5.18 States and stimuli for the microwave oven

- **Activity Diagram**: (Same as above in Interaction Models) Used for depicting the dynamic aspects of the system.
- **Sequence Diagram**: (Same as above in Interaction Models) Used to display the interaction between objects in the chronological order that those interactions occur.
- **Use Case Diagram**: (Same as above in Context Models) Used to represent the functionality of a system from an end-user's point of view.

5.5 Model-Driven Architecture

Model-Driven Architecture (MDA) focuses on creating abstract models that can be transformed into executable code.

- **Class Diagram**: (Same as above in Structural Models) Describes the structure of a system by showing its classes, their attributes, and the relationships among them.
- **State Diagram (State Machine Diagram)**: (Same as above in Behavioral Models) Describes the states an object or interaction may be in, as well as the transitions between states.
- **Sequence Diagram**: (Same as above in Interaction Models) Used to display the interaction between objects in the chronological order that those interactions occur.

used in MDA but no clear name

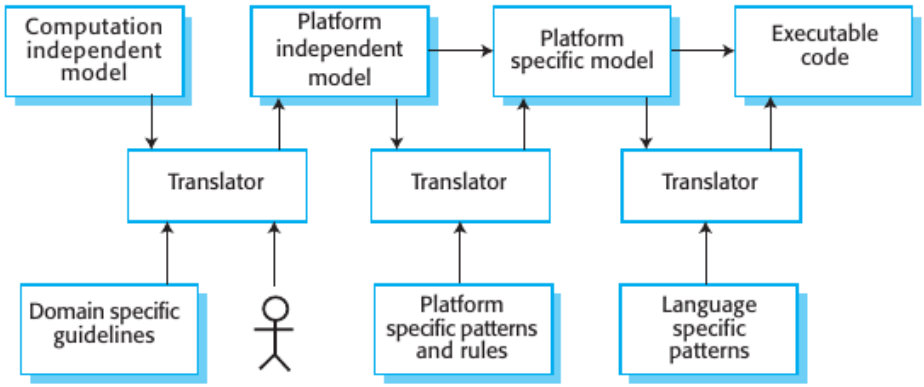

Figure 5.19 MDA transformations

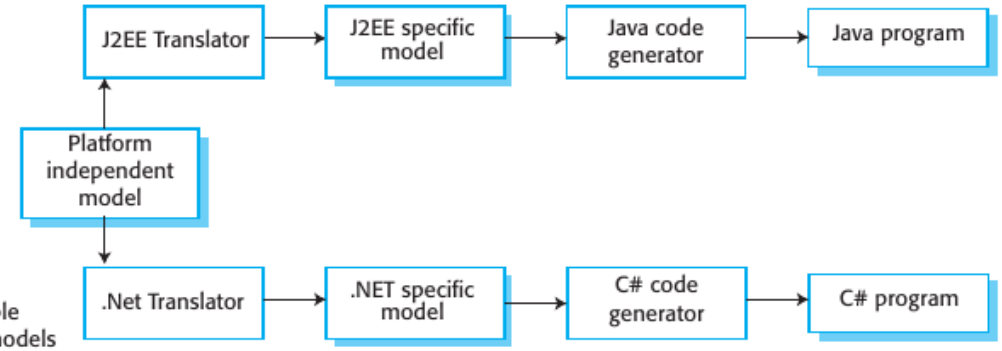

Figure 5.20 Multiple platform-specific models